U.S. DEPARTMENT OF COMMERCE
PATENT AND TRADEMARK OFFICE

CLAIM TO CONVENTION PRIORITY UNDER 35 U.S.C. § 119

Docket Number:	
10191/2173	

Application Number	Filing Date	Examiner	Art Unit
10/034,546	December 28, 2001	To be assigned	3661

Invention Title
METHOD AND DEVICE FOR RECONSTRUCTING THE PROCESS SEQUENCE OF A CONTROL PROGRAM

Inventor(s)
Gabriel WETZEL et al.

Address to:
Assistant Commissioner for Patents
Washington D.C. 20231

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Assistant Commissioner for Patents, Washington, D.C. 20231 on

Date: 6/6/02

Signature: R. No 36,197

A claim to the Convention Priority Date pursuant to 35 U.S.C. § 119 of Application No. 100 65 498.3 filed in the German Patent and Trademark Office on December 28, 2000 is hereby made. To complete the claim to the Convention Priority Date, a certified copy of the priority application is attached.

Dated: 6/6/02

By: _____
Richard L. Mayer (Reg. No. 22,490)

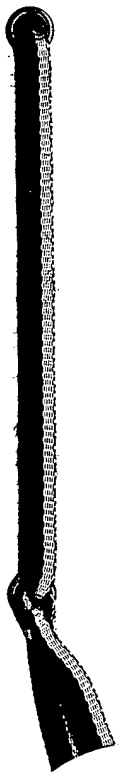
R. No 36,197

KENYON & KENYON
One Broadway
New York, N.Y. 10004
(212) 425-7200 (telephone)
(212) 425-5288 (facsimile)

BUNDESREPUBLIK DEUTSCHLAND

CERTIFIED COPY OF PRIORITY DOCUMENT

Prioritätsbescheinigung über die Einreichung einer Patentanmeldung



Aktenzeichen: 100 65 498.3

Anmeldetag: 28. Dezember 2000

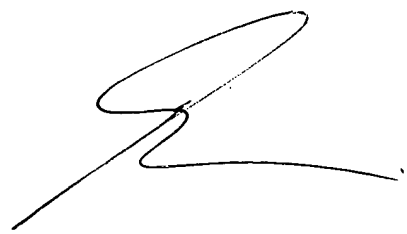
Anmelder/Inhaber: ROBERT BOSCH GMBH, Stuttgart/DE

Bezeichnung: Verfahren und Vorrichtung zur Rekonstruktion des Prozessablaufs eines Steuerprogramms

IPC: G 05 B 19/042

Die angehefteten Stücke sind eine richtige und genaue Wiedergabe der ursprünglichen Unterlagen dieser Patentanmeldung.

München, den 3. Januar 2002
Deutsches Patent- und Markenamt
Der Präsident
Im Auftrag



Dzierzon

A 9161
06/00
EDV-L

R.39072

27.12.2000 wrz
Robert Bosch GmbH, 70469 Stuttgart

Verfahren und Vorrichtung zur Rekonstruktion des Prozessablaufs eines Steuerprogramms

Die vorliegende Erfindung betrifft ein Verfahren und eine Vorrichtung zur Rekonstruktion des Ablaufs von Prozessen eines von einem Rechengerät, insbesondere von einem Mikroprozessor, abgearbeiteten Steuerprogramms aus dem Inhalt einer ersten Tabelle und einer zweiten Tabelle. Das Steuerprogramm ist in mehrere Tasks unterteilt und jede Task umfasst mindestens einen Prozess. Während einer der Rekonstruktion vorangegangenen Abarbeitung des Steuerprogramms wurde in der ersten Tabelle jeweils für eine beendete Task eine Kennung des vor Beginn der beendeten Task zuletzt abgearbeiteten Prozesses abgelegt. In der zweiten Tabelle wurden während der Abarbeitung des Steuerprogramms die Reihenfolge der jeweils beendeten Tasks abgelegt.

Die Erfindung betrifft außerdem ein Speicherelement, insbesondere ein Read-Only-Memory, ein Random-Access-Memory oder ein Flash-Memory. Auf dem Speicherelement ist ein Computerprogramm gespeichert, das auf einem Rechengerät, insbesondere auf einem Mikroprozessor, ablauffähig ist. Schließlich betrifft die vorliegende Erfindung ein Computerprogramm, das auf einem Rechengerät, insbesondere auf einem Mikroprozessor, ablauffähig ist.

Stand der Technik

Ein Steuerprogramm im Sinne der vorliegenden Erfindung dient bspw. zur Steuerung/Regelung von technischen Vorgängen und anderer Funktionen in einem Kraftfahrzeug. Das Steuerprogramm ist auf einem Rechengerät, insbesondere auf einem Mikroprozessor, eines Steuergeräts eines Kraftfahrzeugs ablauffähig. Das Steuerprogramm ist in mehrere Tasks unterteilt und jede Task umfasst mindestens einen Prozess. Den einzelnen Tasks sind unterschiedliche Prioritäten zugeordnet. Das Steuerprogramm kann in einem kooperativen oder in einem preemptiven Modus abgearbeitet werden.

Die Abarbeitung einzelner Tasks eines Steuerprogramms im kooperativen Modus bedeutet, dass bei unterschiedlich priorisierten Tasks eine auszuführende höherpriorisierte Task zu einer Unterbrechung einer aktuell ausgeführten niederpriorisierten Task führt. Anders als im preemptiven Modus, bei dem eine auszuführende höherpriorisierte Task einen aktuell ausgeführten Prozess einer niederpriorisierten Task unterbricht, wartet im kooperativen Modus die höherpriorisierte Task das Ende des aktuell ausgeführten Prozesses der niederpriorisierten Task ab. Erst danach wird die niederpriorisierte Task unterbrochen und die höherpriorisierte Task ausgeführt. Wenn die höherpriorisierte Task fertig ist, wird die niederpriorisierte Task bei dem Prozess fortgesetzt, vor dem sie unterbrochen wurde.

Die Abarbeitung der Tasks eines Steuerprogramms im kooperativen Modus ist aus der DE 195 00 957 A1 bekannt. Die Unterbrechung einer niederpriorisierten Task durch eine höherpriorisierte Task gehört zu den Aufgaben eines Multi-Tasking Betriebssystems. Ein solches Multi-Tasking Betriebssystem, das sowohl den kooperativen Modus als auch den preemptiven Modus bei der Abarbeitung von Steuerprogrammen unterstützt, ist bspw. das

Echtzeitbetriebssystem ERCOS^{EK} von der Firma ETAS Entwicklungs- und Applikationswerkzeuge für elektronische Systeme GmbH & Co. KG, Stuttgart, Deutschland (vgl. ETAS GmbH & Co. KG: ERCOS^{EK} V2.0.0 Manual, Stuttgart, 1998). Auf die DE 195 00 957 A1 und das ERCOS^{EK}-Handbuch wird ausdrücklich Bezug genommen.

Die Laufzeit der Prozesse schwankt je nach Belastung des Rechengeräts. Aus diesem Grund und aufgrund der möglichen, von anderen höherpriorisierten Tasks verursachten Unterbrechungen kann die Reihenfolge der Prozessaufrufe ein und desselben Steuerprogramms bei mehrmaliger Abarbeitung unterschiedlich sein. Das heisst, dass nach der Abarbeitung des Steuerprogramms die genaue Reihenfolge der Prozessaufrufe nicht bekannt ist und auch nicht bspw. für Simulationszwecke rekonstruiert werden kann.

Zur Simulation eines Steuerprogramms oder von Teilen davon (Algorithmus) sind verschiedene Verfahren bekannt. Eine nachträgliche Simulation eines Algorithmus oder des Steuerprogramms mit gemessenen Daten wird als Offline Open Loop Simulation (OOL) bezeichnet. Bei der sogenannten Offline Closed Loop Simulation (OCL) handelt es sich um eine Simulation eines Algorithmus oder des Steuerprogramms mit einem Simulationsmodell in einem geschlossenen Simulationskreis. Die mangelnde Reproduzierbarkeit der Reihenfolge der abgearbeitenden Prozesse führt insbesondere bei einer nachträglichen Simulation des Algorithmus mit gemessenen Daten (OOL) zu erheblichen Schwierigkeiten.

Nach dem Stand der Technik werden Algorithmen, die mit einem Multi-Tasking Betriebssystem gesteuert werden, üblicherweise in einem optimalen Zustand simuliert. Das bedeutet, dass die einzelnen Tasks des Steuerprogramms so aufgerufen werden, dass keine Unterbrechung stattfindet.

Das hat jedoch den Nachteil, dass eine Simulation unter realen Bedingungen nicht möglich ist.

Um eine Simulation eines Steuerprogramms unter realen Bedingungen zu ermöglichen, wird in der DE 100 61 001 ein neuartiges "Verfahren und Steuergerät zur Steuerung von technischen Vorgängen in einem Kraftfahrzeug" vorgeschlagen. Das vorgeschlagene Verfahren beruht auf der Überlegung, während der realen Abarbeitung des Steuerprogramms auf einem Rechengerät den Prozessablauf zu speichern. Es wird insbesondere vorgeschlagen, vor der Abarbeitung des Steuerprogramms jedem Prozess eine eindeutige Kennung zuzuordnen und während der Abarbeitung des Steuerprogramms jeweils die Kennung einer beendeten Task und für die beendete Task die Kennung eines vor Beginn der beendeten Task zuletzt abgearbeiteten Prozesses zu speichern. Dies ist bspw. mit der Speicherung der Kennungen in zwei Tabellen, einer ersten Tabelle (vgl. Figur 3) für die Kennung der vor Beginn der beendeten Tasks zuletzt abgearbeiteten Prozesse und einer zweiten Tabelle (vgl. Figur 4) für die beendeten Tasks, möglich. Alternativ kann für jede Task eine eigene Tabelle vorgesehen werden, in der die Kennungen mit einem entsprechenden Zeitstempel gespeichert werden. Das beschriebene Verfahren zur Speicherung des Prozessablaufs wird nachfolgend anhand der Figuren 2 und 3 näher beschrieben.

In Figur 2 ist der Prozessablauf eines Steuerprogramms dargestellt. Das Steuerprogramm ist in vier Tasks A, B, C, D unterteilt. Die Task A umfasst einen Prozess 111, die Task B Prozesse 212, 222, die Task C Prozesse 313, 323, 333 und die Task D Prozesse 413, 423, 433. Die einzelnen Prozesse der Tasks sind in Figur 2 als Balken dargestellt. Der Task A ist die höchste Priorität zugeordnet, der Task D die niedrigste. Die Prioritäten der Tasks ergeben sich aus der Höhe der dargestellten Balken. Wenn für den in Figur 2

dargestellten Prozessablauf die Kennung eines jeden einzelnen Prozesses abgespeichert würde, müßten alle 22 Prozesskennungen der Prozessablaufliste abgespeichert werden. Die Prozessablaufliste hat den Inhalt: 111, 313, 212, 111, 222, 323, 111, 212, 222, 333, 413, 423, 433, 111, 313, 111, 323, 333, 413, 423, 433, 111.

Um Speicherplatz zu sparen werden deshalb in der ersten Tabelle (vgl. Figur 3) jeweils für eine beendete Task die Kennung eines vor Beginn der beendeten Task zuletzt abgearbeiteten Prozesses gespeichert. Die erste beendete Task des in Figur 2 dargestellten Prozessablaufs ist die Task A (Prozess 111). Der vor Beginn der Task A zuletzt abgearbeitete Prozess ist nicht bekannt. Deshalb ist das erste Element der ersten Tabelle "xxx".

Als nächstes wird die Task C begonnen (Prozess 313), jedoch nicht beendet. Anschließend wird die Task B begonnen (Prozess 212), aber ebenfalls nicht beendet. Die nächste beendete Task ist somit wieder die Task A (Prozess 111). Der vor Beginn der Task A zuletzt abgearbeitete Prozess ist der Prozess 212. Deshalb ist das zweite Element der ersten Tabelle "212".

Als nächstes wird die Task B fortgesetzt (Prozess 222) und beendet. Der vor Beginn der Task B, also vor dem Prozess 212, zuletzt abgearbeitete Prozess ist der Prozess 313. Deshalb ist das dritte Element der ersten Tabelle "313".

Anschließend wird die Task C fortgesetzt (Prozess 323), jedoch nicht beendet. Die nächste beendete Task ist somit wieder die Task A (Prozess 111). Der vor Beginn der Task A zuletzt abgearbeitete Prozess ist der Prozess 323. Deshalb ist das vierte Element der ersten Tabelle "323".

Als nächstes wird die Task B begonnen (Prozess 212) und auch beendet (Prozess 222). Der vor Beginn der Task B zuletzt abgearbeitete Prozess ist der Prozess 111. Deshalb ist das fünfte Element der ersten Tabelle "111".

Anschließend wird die Task C beendet (Prozess 333). Der vor Beginn der Task C, also vor dem Prozess 313, abgearbeitete Prozess ist der Prozess 111. Deshalb ist das sechste Element der ersten Tabelle wieder "111".

Als nächstes wird die Task D begonnen (Prozess 413) und auch beendet (Prozess 433). Der vor Beginn der Task D, also vor dem Prozess 413, abgearbeitete Prozess ist der Prozess 333. Deshalb ist das siebte Element der ersten Tabelle "333". Dieses Verfahren zur Speicherung des Prozessablaufs wird auf den gesamten in Figur 2 dargestellten Prozessablauf angewandt und man erhält die in Figur 3 dargestellte erste Tabelle.

In der zweiten Tabelle (vgl. Figur 4) werden jeweils die beendeten Tasks abgelegt. Die erste beendete Task des Prozessablaufs aus Figur 2 ist die Task A. Danach werden die Tasks C und B begonnen, jedoch nicht beendet. Die nächste beendete Task ist somit wieder die Task A. Danach wird die Task B fortgesetzt und auch beendet. Anschließend wird die Task C fortgesetzt jedoch immer noch nicht beendet. Die nächste beendete Task ist somit wieder die Task A. Danach wird wieder die Task B begonnen und auch beendet. Anschließend wird die Task C wieder fortgesetzt und auch beendet. Das Verfahren wird so lange fortgesetzt bis man als letzten Eintrag in die zweite Tabelle die Task A als letzte beendete Task des in Figur 2 dargestellten Prozessablaufs erhält.

Der vorliegenden Erfindung liegt die Aufgabe zugrunde, den realen Prozessablauf auf eine möglichst einfache Weise aus

dem Inhalt der ersten Tabelle und der zweiten Tabelle vollständig zu reproduzieren.

Zur Lösung dieser Aufgabe schlägt die vorliegende Erfindung ausgehend von dem Verfahren der eingangs genannten Art vor, dass

- zunächst aus dem Inhalt der ersten Tabelle und der zweiten Tabelle eine dritte Tabelle erstellt wird, die jeweils für eine neue Task die Kennung eines vor Beginn der neuen Task zuletzt abgearbeiteten Prozesses enthält, und
- dann aus der dritten Tabelle in Kenntnis des Prozessablaufs der einzelnen Tasks der vollständige Prozessablauf des Steuerprogramms rekonstruiert wird.

Vorteile der Erfindung

Mit dem erfindungsgemäßen Verfahren kann nach der Abarbeitung des Steuerprogramms der Prozessablauf auf einfache Weise anhand der in der ersten und zweiten Tabelle gespeicherten Informationen vollständig reproduziert werden. Bedeutsam ist insbesondere, dass in der ersten Tabelle während der Abarbeitung des Steuerprogramms nicht die Kennungen sämtlicher abgearbeiteter Prozesse, sondern lediglich jeweils für eine beendete Task die Kennung eines vor Beginn der beendeten Task zuletzt abgearbeiteten Prozesses abgelegt wurden. Da die Messungen zum Abspeichern der Reihenfolge der einzelnen Prozesse am Ende der Tasks ausgeführt werden, enthält die zweite Tabelle lediglich Informationen über das Ende der einzelnen Tasks. Mit dem erfindungsgemäßen Verfahren können in der ersten Tabelle fehlende Informationen über den Beginn der einzelnen Tasks rekonstruiert werden.

Der reproduzierte Prozessablauf kann einer Simulation der Algorithmen des Steuerprogramms zugrundegelegt werden.

Dadurch ist eine besonders realitätsnahe Simulation der Algorithmen, insbesondere mit gemessenen Daten nach einer OOL-Simulation, möglich. Aufgrund der Reproduzierbarkeit der simulierten Prozessabläufe können die Messungen und die Simulationsergebnisse miteinander verglichen werden und eine besonders effektive Fehlersuche in dem Steuerprogramm ist möglich.

In der dritten Tabelle ist bis auf eine Ausnahme die vollständige Reihenfolge der einzelnen abgearbeiteten Prozesse abgelegt. Die Ausnahme betrifft eine Folge mehrerer unmittelbar aufeinanderfolgend abgearbeiteter Prozesse derselben Task, wobei für die Folge jeweils nur die letzte Task der Folge in der dritten Tabelle abgelegt ist. Aus dem Inhalt der dritten Tabelle kann somit in Kenntnis des Prozessablaufs der einzelnen Tasks der vollständige Prozessablauf des Steuerprogramms problemlos rekonstruiert werden.

Gemäß einer vorteilhaften Weiterbildung der vorliegenden Erfindung vorgeschlagen, dass zum Erstellen der dritten Tabelle

- zunächst die Kennungen der jeweils letzten Prozesse der in der zweiten Tabelle abgelegten Tasks in der dritten Tabelle abgelegt werden;
- für jede Kennung in der ersten Tabelle geprüft wird, ob der entsprechende Prozess der letzte Prozess seiner Task ist, und
- falls die Kennung dem letzten Prozess ihrer Task entspricht, in der dritten Tabelle kein Eintrag erfolgt; oder
- falls die Kennung nicht dem letzten Prozess ihrer Task entspricht, in der dritten Tabelle die geprüfte Kennung vor die Kennung des ersten in der ersten Tabelle enthaltenen Prozesses der Task abgelegt wird,

die an einer der Position der geprüften Kennung in der ersten Tabelle entsprechenden Position beendet war.

Die geprüfte Kennung wird in der dritten Tabelle vor die Kennung des ersten in der ersten Tabelle enthaltenen Prozesses einer bestimmten Task abgelegt. Der erste in der Tabelle enthaltene Prozess der Task kann der erste Prozess der Task sein. Es ist jedoch auch der Fall denkbar, dass der erste Prozess einer Task garnicht in der ersten Tabelle abgelegt ist, da der erste Prozess während der Abarbeitung des Steuerprogramms nie der vor Beginn einer beendeten Task zuletzt abgearbeitete Prozess ist. In einem solchen Fall wird die geprüfte Kennung dann vor die Kennung des ersten in der ersten Tabelle enthaltenen Prozesses (z. B. des zweiten oder dritten Prozesses) der Task in der dritten Tabelle abgelegt.

Die geprüfte Kennung wird vor die Kennung eines bestimmten Prozesses der Task abgelegt, die an einer der Position der geprüften Kennung in der ersten Tabelle entsprechenden Position beendet war. Mit anderen Worten wird die geprüfte Kennung vor die Kennung eines bestimmten Prozesses der Task abgelegt, die zum Zeitpunkt der Speicherung der geprüften Kennung beendet war.

Gemäß einer bevorzugten Ausführungsform der vorliegenden Erfindung wird vorgeschlagen, dass zum Ermitteln der Kennung des ersten in der ersten Tabelle enthaltenen Prozesses der Task

- eine vierte Tabelle herangezogen wird, in der für jede Task abgelegt ist, ob sie bereits begonnen hat oder nicht, und
- der Inhalt der vierten Tabelle für die Task geprüft wird, die an einer der Position der geprüften Kennung in der ersten Tabelle entsprechenden Position beendet war.

Es wird des weiteren vorgeschlagen, dass in der vierten Tabelle eine Speicherzelle für eine Task gesetzt wird, sobald während der Rekonstruktion des Prozessablaufs auf den Prozess, der als erster von einer anderen Task unterbrochen wird, der Task getroffen wird, und die Speicherzelle für die Task gelöscht wird, sobald während der Rekonstruktion des Prozessablaufs auf den letzten Prozess der Task getroffen wird.

Gemäß einer weiteren bevorzugten Ausführungsform der vorliegenden Erfindung wird vorgeschlagen, dass zum Ermitteln des Prozesses, der als erster von einer anderen Task unterbrochen wird, der Task, die an der der Position der geprüften Kennung in der ersten Tabelle entsprechenden Position beendet war,

- eine fünfte Tabelle herangezogen wird, in der für die in der dritten Tabelle abgelegten Prozesse abgelegt ist, ob die abgelegten Prozesse die Prozesse, die als erste von einer anderen Task unterbrochen werden, der entsprechenden Task sind, und
- der Inhalt der fünften Tabelle für die der Position der geprüften Kennung in der dritten Tabelle vorangehenden Prozesse geprüft wird, ob sie die Prozesse, die als erste von einer anderen Task unterbrochen werden, der Task sind, die an einer der Position der geprüften Kennung in der ersten Tabelle entsprechenden Position beendet war.

Vorteilhafterweise wird in der fünften Tabelle eine Speicherzelle gesetzt, sobald während der Rekonstruktion des Prozessablaufs auf einen in der dritten Tabelle abgelegten Prozess getroffen wird, der der Prozess, der als erster von einer anderen Task unterbrochen wird, der entsprechenden Task ist.

Schließlich wird gemäß noch einer anderen bevorzugten Ausführungsform der vorliegenden Erfindung vorgeschlagen, dass zur Rekonstruktion des vollständigen Prozessablaufs

- die Kennungen der dritten Tabelle in einer Richtung entgegen dem Prozessablauf daraufhin überprüft werden, ob der der geprüften Kennung entsprechende Prozess zu einer Task mit lediglich einem Prozess gehört oder ob es sich bei dem der geprüften Kennung entsprechenden Prozess um den Prozess, der als erster von einer anderen Task unterbrochen wird, der entsprechenden Task handelt;
- die Kennungen der geprüften Prozesse entgegen dem Prozessablauf in einer eindimensionalen siebten Tabelle abgelegt werden, und
- falls der einer geprüften Kennung entsprechende Prozess zu einer Task mit lediglich einem Prozess gehört oder falls es sich bei dem der geprüften Kennung entsprechenden Prozess um den Prozess, der als erster von einer anderen Task unterbrochen wird, der entsprechenden Task handelt, kein Eintrag in die siebte Tabelle erfolgt, oder
- falls der einer geprüften Kennung entsprechende Prozess zu einer Task mit mehreren Prozessen gehört und es sich bei dem der geprüften Kennung entsprechenden Prozess nicht um den Prozess, der als erster von einer anderen Task unterbrochen wird, der entsprechenden Task handelt, in der dritten Tabelle ausgehend von der Position der geprüften Kennung entgegen dem Prozessablauf die Kennung des der geprüften Kennung vorangehenden Prozesses der entsprechenden Task gesucht wird und
- falls vor dem der geprüften Kennung entsprechenden Prozess mindestens ein Prozess fehlt, die Kennung des mindestens einen fehlenden Prozesses in die siebte Tabelle vor den der geprüften Kennung entsprechenden Prozess eingefügt wird.

Von besonderer Bedeutung ist die Realisierung des erfindungsgemäßen Verfahrens in der Form eines Speicherelements. Dabei ist auf dem Speicherelement ein Computerprogramm gespeichert, das auf einem Rechengerät, insbesondere auf einem Mikroprozessor, ablauffähig und zur Ausführung des erfindungsgemäßen Verfahrens geeignet ist. In diesem Fall wird also die Erfindung durch ein auf dem Speicherelement abgespeichertes Computerprogramm realisiert, so dass dieses mit dem Computerprogramm versehene Speicherelement in gleicher Weise die Erfindung darstellt wie das Verfahren, zu dessen Ausführung das Computerprogramm geeignet ist. Als Speicherelement kann insbesondere ein elektrisches Speichermedium zur Anwendung kommen, bspw. ein Read-Only-Memory, ein Random-Access-Memory oder ein Flash-Memory.

Die Erfindung betrifft auch ein Computerprogramm, das zur Ausführung des erfindungsgemäßen Verfahrens geeignet ist, wenn es auf einem Rechengerät, insbesondere auf einem Mikroprozessor, abläuft. Besonders bevorzugt ist dabei, wenn das Computerprogramm auf einem Speicherelement, insbesondere auf einem Flash-Memory, abgespeichert ist.

Als eine weitere Lösung der Aufgabe der vorliegenden Erfindung wird ausgehend von der Vorrichtung zur Rekonstruktion des Ablaufs von Prozessen eines Steuerprogramms der eingangs genannten Art vorgeschlagen, dass die Vorrichtung Mittel zur Ausführung des erfindungsgemäßen Verfahrens aufweist.

Zeichnungen

Weitere Merkmale, Anwendungsmöglichkeiten und Vorteile der Erfindung ergeben sich aus der nachfolgenden Beschreibung von Ausführungsbeispielen der Erfindung, die in der Zeichnung dargestellt sind. Dabei bilden alle beschriebenen

oder dargestellten Merkmale für sich oder in beliebiger Kombination den Gegenstand der Erfindung, unabhängig von ihrer Zusammenfassung in den Patentansprüchen oder deren Rückbeziehung sowie unabhängig von ihrer Formulierung bzw. Darstellung in der Beschreibung bzw. in der Zeichnung. Es zeigen:

Figur 1 ein Ablaufdiagramm eines erfindungsgemäßen Verfahrens gemäß einer bevorzugten Ausführungsform;

Figur 2 einen Prozessablauf eines Steuerprogramms gemäß einer bevorzugten Ausführungsform;

Figur 3 eine erste Tabelle, in der für den Prozessablauf aus Figur 2 jeweils für eine beendete Task die Kennung eines vor Beginn der beendeten Task zuletzt abgearbeiteten Prozesses abgelegt ist;

Figur 4 eine zweite Tabelle, in der für den Prozessablauf aus Figur 2 die Reihenfolge der jeweils beendeten Tasks abgelegt ist;

Figur 5 eine dritte Tabelle zu Beginn des erfindungsgemäßen Verfahrens, in der die Kennungen der jeweils letzten Prozesse der in der zweiten Tabelle aus Figur 4 abgelegten Tasks abgelegt sind;

Figur 6 die dritte Tabelle aus Figur 5 am Ende des erfindungsgemäßen Verfahrens, in der jeweils für eine neue Task die Kennung eines vor Beginn der neuen Task zuletzt abgearbeiteten Prozesses abgelegt ist;

Figur 7 eine vierte Tabelle, in der während der Ausführung des erfindungsgemäßen Verfahrens für jede Task abgelegt wird, ob sie bereits begonnen hat oder nicht;

Figur 8 eine fünfte Tabelle, in der während der Ausführung des erfindungsgemäßen Verfahrens abgelegt wird, ob die in der dritten Tabelle abgelegten Prozesse die ersten Prozesse der entsprechenden Task sind;

Figur 9 Inhalt der dritten Tabelle aus Figur 6 am Ende des erfindungsgemäßen Verfahrens;

Figur 10 eine siebte Tabelle mit einem aus der ersten Tabelle aus Figur 3 und der zweiten Tabelle aus Figur 4 rekonstruierten Prozessablauf des Steuerprogramms; und

Figur 11 eine erfindungsgemäße Vorrichtung gemäß einer bevorzugten Ausführungsform.

Beschreibung der Ausführungsbeispiele

Auf einem Rechengerät, insbesondere auf einem Mikroprozessor, eines Steuergeräts können Steuerprogramme zur Steuerung von technischen Vorgängen insbesondere in einem Kraftfahrzeug abgearbeitet werden. Die Steuerprogramme können in mehrere Tasks unterteilt sein, wobei jede Task wiederum mindestens einen Prozess umfasst. Eine Task wird zu einem bestimmten Zeitpunkt oder regelmäßig mit einer bestimmten Abtastzeit aufgerufen und kann in einem kooperativen oder in einem preemptiven Modus abgearbeitet werden. Jeder Task ist eine bestimmte Priorität zugeordnet. Wenn während der Abarbeitung des Steuerprogramms zwei Tasks gleichzeitig ausgeführt werden

Unterbrechungen kann die Reihenfolge der Prozessaufrufe bei einer mehrmaligen Ausführung ein und desselben Steuerprogramms unterschiedlich sein. Nach der Abarbeitung des Steuerprogramms ist also die Reihenfolge, in der die einzelnen Prozesse der Tasks aufgerufen wurden, nicht bekannt.

Um eine Simulation eines Steuerprogramms unter realen Bedingungen zu ermöglichen, wird in einer weiteren Patentanmeldung der Anmelderin der vorliegenden Patentanmeldung ein neuartiges "Verfahren und Steuergerät zur Steuerung von technischen Vorgängen in einem Kraftfahrzeug" vorgeschlagen. Das dort vorgeschlagene Verfahren beruht auf der Überlegung, während der realen Abarbeitung des Steuerprogramms auf einem Rechengerät den Prozessablauf zu speichern. Es wird insbesondere vorgeschlagen, vor der Abarbeitung des Steuerprogramms jedem Prozess eine eindeutige Kennung zuzuordnen und während der Abarbeitung des Steuerprogramms lediglich jeweils für eine beendete Task die Kennung eines vor Beginn der beendeten Task zuletzt abgearbeiteten Prozesses in einer ersten Tabelle ProcMeasArray zu speichern. Das Verfahren zur Speicherung des Prozessablaufs bildet den Ausgangspunkt für die vorliegende Patentanmeldung und ist in der Beschreibungseinleitung ausführlich beschrieben, worauf Bezug genommen wird.

Die vorliegende Patentanmeldung betrifft ein Verfahren zur Rekonstruktion des realen Prozessablaufs des Steuerprogramms aus dem Inhalt der ersten Tabelle ProcMeasArray (vgl. Figur 3). Zur besseren Zuordnung der einzelnen Elemente der Tabellen ist neben den Tabellen aus den Figuren 3 bis 10 eine Laufvariable, ein sog. LoopIndex, aufgetragen.

Das Verfahren beginnt in einem Funktionsblock 1 in Figur 1a. Dann wird in einem Funktionsblock 2 zunächst ein Initialisierungsschritt ausgeführt, in dem alle für das Verfahren notwendigen Größen initialisiert werden und eine zweite Tabelle TaskMeasArray (vgl. Figur 4) ausgewertet wird. In einem Funktionsblock 3 wird dann ein Rekonstruktionsalgorithmus abgearbeitet, um die Unterbrechnungen der Tasks A, B, C, D rekonstruieren zu können. Der Algorithmus basiert hauptsächlich auf der Auswertung der ersten Tabelle ProcMeasArray. Für jeden Aufruf einer Folge von mindestens einem Prozess derselben Task wird die Kennung des letzten abgearbeiteten Prozesses in der dritten Tabelle SimArrayIdent abgelegt. In einem Funktionsblock 4 werden dann anhand der dritten Tabelle SimArrayIdent alle Prozesse in der richtigen Reihenfolge aufgerufen. In einem Funktionsblock 5 ist das erfindungsgemäße Verfahren beendet.

Anhand Figur 1b wird der Initialisierungsschritt 2 näher erläutert. In einem Funktionsblock 6 wird eine dritte Tabelle SimArrayIdent (vgl. Figur 5) der Dimension (Anzahl gemessener Taskaufrufe * Taskanzahl) initialisiert. Die Anzahl der gemessenen Taskaufrufe ist gleich der Dimension der ersten Tabelle ProcMeasArray, also gleich zwölf. Die Taskanzahl ist gleich vier (Task A, Task B, Task C, Task D). In der ersten Zeile der dritten Tabelle SimArrayIdent wird jeweils die Kennung des letzten Prozesses 111, 222, 333, 433 jeder in der zweiten Tabelle TaskMeasArray abgelegten Task A, B, C, D abgelegt. In der zweiten Tabelle TaskMeasArray wurde während der Abarbeitung des Steuerprogramms für den Prozessablauf aus Figur 2 die Reihenfolge der jeweils beendeten messenden Tasks A, B, C, D abgelegt.

In einem Funktionsblock 7 wird eine vierte Tabelle BegunTask (vgl. Figur 7) der Dimension (Taskanzahl * 1)

initialisiert. In der vierten Tabelle BegunTask wird während der Ausführung des Rekonstruktionsalgorithmus eine Speicherzelle für eine Task A, B, C, D gesetzt, sobald auf den ersten in der ersten Tabelle ProcMeasArray enthaltenen Prozess 111; 212; 313; 413 einer Task A, B, C, D getroffen wird. Die Speicherzelle wird für die Task A, B, C, D gelöscht, sobald auf den letzten Prozess 111; 222; 333; 433 einer Task A, B, C, D getroffen wird. Der Inhalt der vierten Tabelle BegunTask wird auf Null gesetzt, da die dritte Tabelle SimArrayIdent zunächst nur beendete Tasks A, B, C, D (die letzten Prozesse 111, 222, 333, 433 der Tasks A, B, C, D) beinhaltet.

In einem Funktionsblock 8 wird des weiteren eine fünfte Tabelle SimArrayBeginning (vgl. Figur 8) der Dimension (Anzahl gemessener Taskaufrufe * Taskanzahl) initialisiert. wird während der Ausführung des Rekonstruktionsalgorithmus der Anfang einer Task in der dritten Tabelle SimArrayIdent gefunden, wird an der gleichen Position (gleiche Zeile und Spalte) in der fünften Tabelle SimArrayBeginning eine entsprechende Speicherzelle auf Eins gesetzt. Wird während des Rekonstruktionsalgorithmus mit Hilfe der vierten Tabelle BegunTask festgestellt, dass die zu einem in der ersten Tabelle ProcMeasArray abgelegten gemessenen Prozess gehörende Task bereits begonnen hat, wird die Speicherzelle in der ersten Zeile dieser Spalte der fünften Tabelle SimArrayBeginning zurückgesetzt. Die erste Zeile der fünften Tabelle SimArrayBeginning wird im Rahmen des Initialisierungsschritts 2 auf Eins gesetzt.

In einem Funktionsblock 9 wird eine sechste Tabelle ColProcNum der Dimension (Anzahl gemessener Taskaufrufe * 1) initialisiert. In der sechste Tabelle ColProcNum wird während der Ausführung des Rekonstruktionsalgorithmus für jede Spalte der Zeilenindex der letzten Speicherzelle, die

ungleich Null ist, der fünften Tabelle SimArrayBeginning abgelegt.

Anhand Figur 1c wird der Rekonstruktionsalgorithmus 3 näher erläutert. Es wird beginnend bei dem ersten Element der ersten Tabelle ProcMeasArray nacheinander jedes Element, d.h. jeder gemessene Prozess, überprüft. In einem Funktionsblock 10 wird der LoopIndex auf Null gesetzt (LoopIndex=0). In einem Abfrageblock 11 wird überprüft ob der dem aktuellen LoopIndex entsprechende Prozess 111 dem letzten Prozess 111 der dem Prozess 111 zugeordneten Task A entspricht. Da dies der Fall ist, erfolgt kein Eintrag in der dritten Tabelle SimArrayIdent, und es wird zu einem Funktionsblock 12 verzweigt, wo der LoopIndex um Eins erhöht wird (LoopIndex=1).

In dem Abfrageblock 11 wird dann der nächste Prozess 212 der ersten Tabelle ProcMeasArray überprüft. Da der Prozess 212 nicht dem letzten Prozess 222 der dem Prozess 212 zugeordneten Task B entspricht, wird zu einem Funktionsblock 13 verzweigt, wo die dem aktuellen LoopIndex entsprechende messende Task anhand der zweiten Tabelle TaskMeasArray ermittelt wird (Task A). Anschließend wird in einem Funktionsblock 14 der Beginn, d.h. der erste in der ersten Tabelle ProcMeasArray enthaltene Prozess, der messenden Task ermittelt. Zur Ermittlung des Beginns der messenden Task werden die vierte Tabelle BegunTask (vgl. Figur 7), die fünfte Tabelle SimArrayBeginning und die sechste Tabelle ColProcNum herangezogen. Selbstverständlich kann der Beginn der messenden Task auch auf eine andere Weise ermittelt werden.

Für die Task A ist die Ermittlung des Beginns der messenden Task sehr einfach, da der Prozess 111 immer auch den Beginn der Task A darstellt. Dann wird der aktuelle überprüfte Prozess 212 in einem Funktionsblock 15 vor den Beginn der

messenden Task A in der dritten Tabelle SimArrayIdent abgelegt (vgl. Figur 6). Schließlich wird in einem weiteren Abfrageblock 16 überprüft, ob alle in der ersten Tabelle ProcMeasArray abgelegten Prozesse überprüft worden sind. Falls nein, wird zu dem Funktionsblock 12 verzweigt, der LoopIndex um Eins erhöht (LoopIndex=2) und der nächste Prozess der ersten Tabelle ProcMeasArray überprüft. Falls ja, wird zu dem Aufruf 4 des Prozessablaufs in der richtigen Reihenfolge verzweigt.

In Fortsetzung des Rekonstruktionsalgorithmus 3 wird für den LoopIndex=2 der Prozess 313 der ersten Tabelle ProcMeasArray überprüft. In dem Abfrageblock 11 wird festgestellt, dass der Prozess 313 nicht dem letzten Prozess 333 der dem Prozess 313 zugeordneten Task C entspricht. In dem Funktionsblock 13 wird die Task B als die dem aktuellen LoopIndex=2 entsprechende messende Task ermittelt. Anschließend wird in dem Funktionsblock 14 der Beginn, d.h. der erste in der ersten Tabelle ProcMeasArray enthaltene Prozess 212, der messenden Task B ermittelt. Dann wird der aktuelle überprüfte Prozess 313 in dem Funktionsblock 15 vor den Beginn der messenden Task B in der dritten Tabelle SimArrayIdent abgelegt (vgl. Figur 6). Der Rekonstruktionsalgorithmus wird so lange fortgesetzt bis sich die dritte Tabelle SimArrayIdent mit dem in Figur 6 dargestellten Inhalt ergibt.

Anhand Figur 1d wird der Aufruf 4 aller Prozesse in der richtigen Reihenfolge näher erläutert. Die in der dritten Tabelle SimArrayIdent abgelegten Prozesse werden in der in Figur 6 durch Pfeile gekennzeichneten Reihenfolge aus der dritten Tabelle SimArrayIdent ausgelesen. Man erhält so eine Folge von vor Beginn einer neuen Task A, B, C, D zuletzt abgearbeiteten Prozessen 111; 212, 222; 313, 323, 333; 413, 423, 433. In der dritten Tabelle SimArrayIdent ist für eine Folge mehrerer Prozesse 212, 222; 313, 323,

333; 413, 423, 433 derselben Task B; C; D lediglich der jeweils letzte Prozess 222; 333; 433 der Task B; C; D abgelegt. Die fehlenden Prozesse werden in dem Funktionsblock 4 ergänzt.

Die einzelnen Kennungen der dritten Tabelle SimArrayIdent werden ausgehend von der letzten Kennung in einer Richtung entgegen dem Prozessablauf überprüft. Zum besseren Verständnis ist in Figur 9 der Inhalt der dritten Tabelle SimArrayIdent in einer eindimensionalen Tabelle mit einem mit dem Prozessablauf zunehmenden Zeiger LoopIndex neben den entsprechenden Tabellenelementen dargestellt. In einem Funktionsblock 17 wird der Zeiger LoopIndex auf die Anzahl der in der dritten Tabelle SimArrayIdent abgelegten Kennungen abzüglich Eins gesetzt (LoopIndex=15). Über den Zeiger LoopIndex wird die zu prüfende Kennung ausgewählt. Zunächst wird die Kennung des der geprüften Kennung entsprechenden Prozesses 111 in einer eindimensionalen siebten Tabelle (vgl. Figur 10) als letztes Element abgelegt. In einem Abfrageblock 18 wird überprüft, ob der Prozess 111 einer Task mit lediglich einem Prozess zugeordnet ist. Der Prozess 111 ist der Task A zugeordnet, die lediglich einen Prozess 111 umfasst. Es wird zu dem Funktionsblock 24 verzweigt, in dem ausgehend von dem der geprüften Kennung entsprechenden Prozess alle Prozesse der betrachteten Task bis zu dem ersten Prozess der Task vor den geprüften Prozess in die siebte Tabelle eingefügt werden. Im vorliegenden Fall werden keine weiteren Kennungen von Prozessen in der siebten Tabelle abgelegt. Es wird zu einem Funktionsblock 19 verzweigt, wo der LoopIndex um Eins erniedrigt wird (LoopIndex=14). Anschließend wird in einem weiteren Abfrageblock 20 überprüft, ob alle in der dritten Tabelle SimArrayIdent abgelegten Kennungen überprüft worden sind (LoopIndex<0?).

Zunächst wird die Kennung des der geprüften Kennung entsprechenden Prozesses 433 in der siebten Tabelle (vgl. Figur 10) als vorletztes Element abgelegt. Dann wird in dem Abfrageblock 18 überprüft, ob der Prozess 433 einer Task D mit lediglich einem Prozess 413, 423, 433 zugeordnet ist. Das ist nicht der Fall, und es wird zu einem weiteren Abfrageblock 21 verzweigt, wo überprüft wird, ob es sich bei dem der geprüften Kennung entsprechenden Prozess um den Prozess, der als erster von einer anderen Task unterbrochen wird, der entsprechenden Task handelt. Anhand der fünften Tabelle wird ermittelt, dass der Prozess 433 der erste von einer anderen Task unterbrochene Prozess ist. Es wird zu dem Funktionsblock 24 verzweigt, in dem ausgehend von dem der geprüften Kennung entsprechenden Prozess 433 alle Prozesse 423, 413 der betrachteten Task D bis zu dem ersten Prozess 413 der Task D vor den geprüften Prozess 433 in der siebten Tabelle abgelegt werden. Dann wird zu dem Funktionsblock 19 verzweigt, wo der LoopIndex wieder um Eins erniedrigt wird (LoopIndex=13). Anschließend wird in dem Abfrageblock 20 wieder überprüft, ob alle in der dritten Tabelle SimArrayIdent abgelegten Kennungen überprüft worden sind (LoopIndex<0?).

Zunächst wird die Kennung des der geprüften Kennung entsprechenden Prozesses 333 in der siebten Tabelle (vgl. Figur 10) abgelegt. Dann wird in dem Abfrageblock 18 überprüft, ob der Prozess 333 einer Task C mit lediglich einem Prozess 313, 323, 333 zugeordnet ist. Das ist nicht der Fall, und es wird zu einem weiteren Abfrageblock 21 verzweigt, wo überprüft wird, ob es sich bei dem der geprüften Kennung entsprechenden Prozess 333 um den Prozess, der als erster von einer anderen Task unterbrochen wird, der entsprechenden Task handelt.

In diesem Fall ist der Prozess 333 jedoch nicht der erste Prozess 313 der Task C, der von einer anderen Task

unterbrochen wird. Deshalb wird zu einem Funktionsblock 22 verzweigt, in dem die dritte Tabelle SimArrayIdent entgegen dem Prozessablauf nach einem dem aktuell überprüften Prozess vorangehenden Prozess durchsucht und mit Hilfe der fünten Tabelle der erste Prozess der Task C ermittelt wird, der von einer anderen Task unterbrochen wird. Anschließend werden in einem Funktionsblock 23 diejenigen Prozesse vor den überprüften Prozess 333 in die siebte Tabelle (vgl. Figur 10) eingefügt, die zwischen dem überprüften Prozess 333 und dem in dem Funktionsblock 23 ermittelten ersten von einer anderen Task unterbrochenen Prozess 313 der Task C liegen. Im vorliegenden Fall wird also lediglich die Kennung des Prozesses 323 eingefügt.

Das in Figur 1d dargestellte Verfahren wird so lange fortgesetzt, bis sämtliche in der dritten Tabelle SimArrayIdent abgelegten Kennungen überprüft worden sind (LoopIndex<0 in dem Abfrageblock 20). Man erhält dann den in der siebten Tabelle abgelegten vollständigen reproduzierten Prozessablauf des Steuerprogramms (vgl. Figur 10).

In Figur 11 ist eine erfindungsgemäße Vorrichtung mit dem Bezugszeichen 30 bezeichnet. Die Vorrichtung 30 weist ein Rechengerät, insbesondere einen Mikroprozessor 31 auf, auf dem ein Computerprogramm ablauffähig ist. Die Vorrichtung 30 umfasst des Weiteren ein Speicherelement 32, auf dem das Computerprogramm gespeichert ist. Über eine Datenverbindung 33, die bspw. als eine Bus-Leitung ausgebildet ist, wird das von dem Mikroprozessor 31 auszuführende Computerprogramm oder Teile davon zu dem Mikroprozessor 31 übertragen und dort abgearbeitet. Über geeignete Schnittstellen 34 werden dem Mikroprozessor 31 während der Abarbeitung des Computerprogramms Messgrößen zugeführt und in dem Mikroprozessor 31 verarbeitet. Die Messgrößen können aber auch in dem Speicherelement 32 gespeichert werden und

während der Abarbeitung des Computerprogramms an den Mikroprozessor 31 übertragen werden. Die Messgrößen umfassen bspw. auch Informationen über den Prozessablauf, insbesondere die Reihenfolge der Abarbeitung der Prozesse, eines Steuerprogramms. Diese Informationen sind in einer ersten Tabelle ProcMeasArray abgelegt. Durch die Abarbeitung des Computerprogramms auf dem Mikroprozessor 31 kann das oben beschriebene erfindungsgemäße Verfahren ausgeführt werden.

27.12.2000
Robert Bosch GmbH, 70469 Stuttgart

Ansprüche

1. Verfahren zur Rekonstruktion des Ablaufs von Prozessen eines von einem Rechengerät, insbesondere von einem Mikroprozessor, abgearbeiteten Steuerprogramms aus dem Inhalt einer ersten Tabelle (ProcMeasArray) und einer zweiten Tabelle (TaskMeasArray), wobei das Steuerprogramm in mehrere Tasks (A, B, C, D) unterteilt ist und jede Task (A, B, C, D) mindestens einen Prozess (111; 212, 222; 313, 323, 333; 413, 423, 433) umfasst und während der Abarbeitung des Steuerprogramms in der ersten Tabelle (ProcMeasArray) jeweils für eine beendete Task (A, B, C, D) eine Kennung des vor Beginn der beendeten Task (A, B, C, D) zuletzt abgearbeiteten Prozesses (111; 212, 222; 313, 323, 333; 413, 423, 433) und in der zweiten Tabelle (TaskMeasArray) die Reihenfolge der jeweils beendeten Tasks (A, B, C, D) abgelegt wurde, **dadurch gekennzeichnet**, dass

- zunächst aus dem Inhalt der ersten Tabelle (ProcMeasArray) und der zweiten Tabelle (TaskMeasArray) eine dritte Tabelle (SimArrayIdent) erstellt wird, die jeweils für eine neue Task (A, B, C, D) die Kennung des vor Beginn der neuen Task zuletzt abgearbeiteten Prozesses (111; 212, 222; 313, 323, 333; 413, 423, 433) enthält, und
- dann aus der dritten Tabelle (SimArrayIdent) in Kenntnis der Reihenfolge der Abarbeitung der Prozesse (111; 212, 222; 313, 323, 333; 413, 423, 433) der einzelnen Tasks (A, B, C, D) der vollständige Prozessablauf des Steuerprogramms rekonstruiert wird.

2. Verfahren nach Anspruch 1, dadurch gekennzeichnet, dass zum Erstellen der dritten Tabelle (SimArrayIdent)

- zunächst die Kennungen der jeweils letzten Prozesse (111; 222; 333; 433) der in der zweiten Tabelle (TaskMeasArray) abgelegten Tasks (A, B, C, D) in der dritten Tabelle (SimArrayIdent) abgelegt werden;
- für jede Kennung in der ersten Tabelle (ProcMeasArray) geprüft wird, ob der entsprechende Prozess (111; 212, 222; 313, 323, 333; 413, 423, 433) der letzte Prozess (111; 222; 333; 433) seiner Task (A, B, C, D) ist, und
- falls die Kennung dem letzten Prozess (111; 222; 333; 433) ihrer Task (A, B, C, D) entspricht, in der dritten Tabelle (SimArrayIdent) kein Eintrag erfolgt; oder
- falls die Kennung nicht dem letzten Prozess (111; 222; 333; 433) ihrer Task (A, B, C, D) entspricht, in der dritten Tabelle (SimArrayIdent) die geprüfte Kennung vor die Kennung des ersten in der ersten Tabelle (ProcMeasArray) enthaltenen Prozesses (111; 212; 313; 413) der Task (A, B, C, D) abgelegt wird, die an einer der Position der geprüften Kennung in der ersten Tabelle (ProcMeasArray) entsprechenden Position beendet war.

3. Verfahren nach Anspruch 2, dadurch gekennzeichnet, dass zum Ermitteln der Kennung des ersten in der ersten Tabelle (ProcMeasArray) enthaltenen Prozesses (111; 212; 313; 413) der Task (A, B, C, D)

- eine vierte Tabelle (BegunTask) herangezogen wird, in der für jede Task (A, B, C, D) abgelegt ist, ob sie bereits begonnen hat oder nicht, und
- der Inhalt der vierten Tabelle (BegunTask) für die Task (A, B, C, D) geprüft wird, die an einer der Position der geprüften Kennung in der ersten Tabelle (ProcMeasArray) entsprechenden Position beendet war.

4. Verfahren nach Anspruch 3, dadurch gekennzeichnet, dass in der vierten Tabelle (BegunTask) eine Speicherzelle für eine Task (A, B, C, D) gesetzt wird, sobald während der Rekonstruktion des Prozessablaufs auf den ersten von einer anderen Task unterbrochenen Prozess (111; 212; 313; 413) der Task (A, B, C, D) getroffen wird, und die Speicherzelle für die Task (A, B, C, D) gelöscht wird, sobald während der Rekonstruktion des Prozessablaufs auf den letzten Prozess (111; 222; 333; 433) der Task (A, B, C, D) getroffen wird.

5. Verfahren nach einem der Ansprüche 2 bis 4, dadurch gekennzeichnet, dass zum Ermitteln des ersten von einer anderen Task unterbrochenen Prozesses (111; 212; 313; 413) der Task (A, B, C, D), die an der der Position der geprüften Kennung in der ersten Tabelle (ProcMeasArray) entsprechenden Position beendet war,

- eine fünfte Tabelle (SimArrayBeginning) herangezogen wird, in der für die in der dritten Tabelle (SimArrayIdent) abgelegten Prozesse (111; 212, 222; 313, 323, 333; 413, 423, 433) abgelegt ist, ob die abgelegten Prozesse (111; 212, 222; 313, 323, 333; 413, 423, 433) die ersten von einer anderen Task unterbrochenen Prozesse (111; 212; 313; 413) der entsprechenden Task (A, B, C, D) sind, und
- der Inhalt der fünften Tabelle (SimArrayBeginning) für die der Position der geprüften Kennung in der dritten Tabelle (SimArrayIdent) vorangehenden Prozesse (111; 212, 222; 313, 323, 333; 413, 423, 433) geprüft wird, ob sie die ersten von einer anderen Task unterbrochenen Prozesse der Task (A, B, C, D) sind, die an einer der Position der geprüften Kennung in der ersten Tabelle (ProcMeasArray) entsprechenden Position beendet war.

6. Verfahren nach Anspruch 5, dadurch gekennzeichnet, dass in der fünften Tabelle (SimArrayBeginning) eine

Speicherzelle gesetzt wird, sobald während der Rekonstruktion des Prozessablaufs auf einen in der dritten Tabelle (SimArrayIdent) abgelegten Prozess (111; 212, 222; 313, 323, 333; 413, 423, 433) getroffen wird, der der erste von einer anderen Task unterbrochene Prozess (111; 212; 313; 413) der entsprechenden Task (A, B, C, D) ist.

7. Verfahren nach einem der Ansprüche 1 bis 6, dadurch gekennzeichnet, dass zur Rekonstruktion des vollständigen Prozessablaufs

- die Kennungen der dritten Tabelle (SimArrayIdent) in einer Richtung entgegen dem Prozessablauf daraufhin überprüft werden, ob der der geprüften Kennung entsprechende Prozess (111; 212, 222; 313, 323, 333; 413, 423, 433) zu einer Task (A) mit lediglich einem Prozess (111) gehört oder ob es sich bei dem der geprüften Kennung entsprechenden Prozess (111; 212, 222; 313, 323, 333; 413, 423, 433) um den ersten in der ersten Tabelle (ProcMeasArray) enthaltenen Prozess (111; 212; 313; 413) der entsprechenden Task (A, B, C, D) handelt;
- die Kennungen der geprüften Prozesse entgegen dem Prozessablauf in einer eindimensionalen siebten Tabelle abgelegt werden, und
- falls ein der geprüften Kennung entsprechende Prozess (111; 212, 222; 313, 323, 333; 413, 423, 433) zu einer Task (A) mit lediglich einem Prozess (111) gehört oder falls es sich bei dem der geprüften Kennung entsprechenden Prozess (111; 212, 222; 313, 323, 333; 413, 423, 433) um den ersten von einer anderen Task unterbrochenen Prozess (111; 212; 313; 413) der entsprechenden Task (A, B, C, D) handelt, kein Eintrag in die siebte Tabelle erfolgt, oder
- falls ein der geprüften Kennung entsprechende Prozess (111; 212, 222; 313, 323, 333; 413, 423, 433) zu einer Task (B, C, D) mit mehreren Prozessen (212, 222; 313,

323, 333; 413, 423, 433) gehört und es sich bei dem der geprüften Kennung entsprechenden Prozess (111; 212, 222; 313, 323, 333; 413, 423, 433) nicht um den ersten von einer anderen Task unterbrochenen Prozess (212; 313; 413) der entsprechenden Task (B, C, D) handelt, in der dritten Tabelle (SimArrayIdent) ausgehend von der geprüften Position entgegen dem Prozessablauf die Kennung des der geprüften Kennung vorangehenden Prozesses (212; 313, 323; 413, 423) der entsprechenden Task (B, C, D) gesucht wird und

- falls vor dem der geprüften Kennung entsprechenden Prozess (222; 323, 333; 423, 433) mindestens ein Prozess (212; 313, 323; 413, 423) fehlt, die Kennung des mindestens einen fehlenden Prozesses (212; 313, 323; 413, 423) in die siebte Tabelle vor den der geprüften Kennung entsprechenden Prozess (222; 323, 333; 423, 433) eingefügt wird.

8. Speicherelement (32), insbesondere Read-Only-Memory, Random-Access-Memory oder Flash-Memory, auf dem ein Computerprogramm gespeichert ist, das auf einem Rechengerät, insbesondere auf einem Mikroprozessor, ablauffähig und zur Ausführung eines Verfahrens nach einem der Ansprüche 1 bis 7 geeignet ist.

9. Computerprogramm, dadurch gekennzeichnet, dass das Computerprogramm zur Ausführung eines Verfahrens nach einem der Ansprüche 1 bis 7 geeignet ist, wenn es auf einem Rechengerät, insbesondere auf einem Mikroprozessor, abläuft.

10. Computerprogramm nach Anspruch 9, dadurch gekennzeichnet, dass das Computerprogramm auf einem Speicherelement, insbesondere auf einem Flash-Memory, abgespeichert ist.

11. Vorrichtung zur Rekonstruktion des Ablaufs von Prozessen eines von einem Rechengerät, insbesondere von einem Mikroprozessor, abgearbeiteten Steuerprogramms aus dem Inhalt einer ersten Tabelle (ProcMeasArray) und einer zweiten Tabelle (TaskMeasArray), wobei das Steuerprogramm in mehrere Tasks (A, B, C, D) unterteilt ist und jede Task (A, B, C, D) mindestens einen Prozess (111; 212, 222; 313, 323, 333; 413, 423, 433) umfasst und während der Abarbeitung des Steuerprogramms in der ersten Tabelle (ProcMeasArray) jeweils für eine beendete Task (A, B, C, D) eine Kennung eines vor Beginn der beendeten Task (A, B, C, D) zuletzt abgearbeiteten Prozesses (111; 212, 222; 313, 323, 333; 413, 423, 433) und in der zweiten Tabelle (TaskMeasArray) die Reihenfolge der jeweils beendeten Tasks (A, B, C, D) abgelegt ist, **dadurch gekennzeichnet**, dass die Vorrichtung Mittel zur Ausführung eines Verfahrens nach einem der Ansprüche 1 bis 7 aufweist.

27.12.2000
Robert Bosch GmbH, 70469 Stuttgart

Verfahren und Vorrichtung zur Rekonstruktion des Prozessablaufs eines Steuerprogramms

Zusammenfassung

Die Erfindung betrifft ein Verfahren und eine Vorrichtung zur Rekonstruktion des Ablaufs von Prozessen eines von einem Rechengerät, insbesondere von einem Mikroprozessor, abgearbeiteten Steuerprogramms aus dem Inhalt einer ersten Tabelle (ProcMeasArray) und einer zweiten Tabelle (TaskMeasArray). Das Steuerprogramm ist in mehrere Tasks (A, B, C, D) unterteilt und jede Task (A, B, C, D) umfasst mindestens einen Prozess (111; 212, 222; 313, 323, 333; 413, 423, 433). Während der Abarbeitung des Steuerprogramms wurde in der ersten Tabelle (ProcMeasArray) jeweils für eine beendete Task (A, B, C, D) eine Kennung eines vor Beginn der beendeten Task (A, B, C, D) zuletzt abgearbeiteten Prozesses (111; 212, 222; 313, 323, 333; 413, 423, 433) abgelegt. In der zweiten Tabelle wurde die Reihenfolge der jeweils beendeten Tasks (A, B, C, D) abgelegt. Um auf möglichst einfache Weise eine zuverlässige Rekonstruktion des Prozessablaufs realisieren zu können, wird vorgeschlagen, dass

- zunächst aus dem Inhalt der ersten Tabelle (ProcMeasArray) und der zweiten Tabelle (TaskMeasArray) eine dritte Tabelle (SimArrayIdent) erstellt wird, die jeweils für eine neue Task (A, B, C, D) die Kennung eines vor Beginn der neuen Task zuletzt abgearbeiteten Prozesses (111; 212, 222; 313, 323, 333; 413, 423, 433) enthält, und
- dann aus der dritten Tabelle (SimArrayIdent) in Kenntnis des Prozessablaufs der einzelnen Tasks (A, B, C, D) der vollständige Prozessablauf des Steuerprogramms rekonstruiert wird. (Figur 1a)

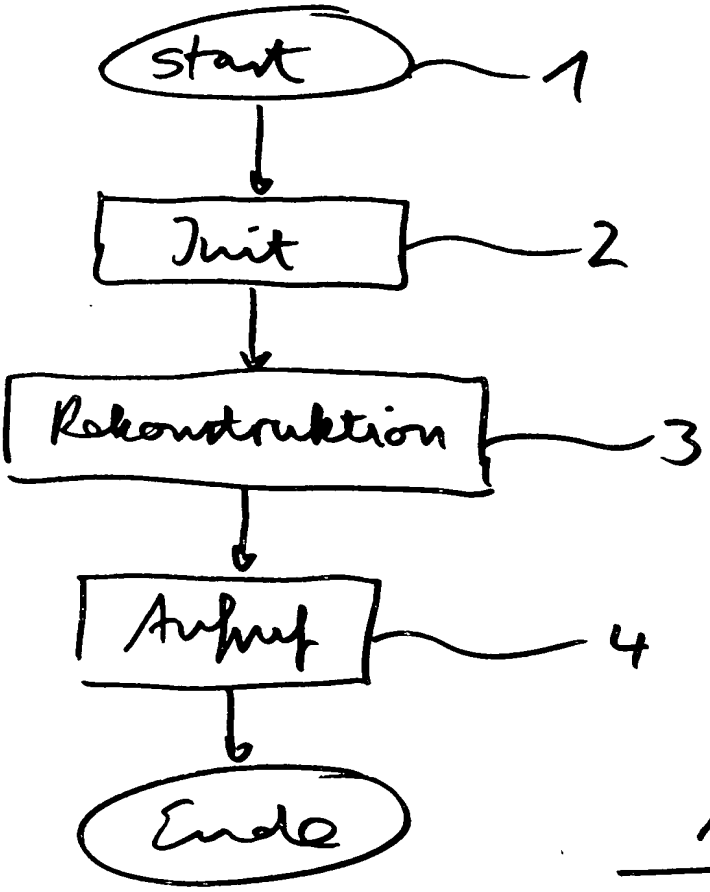

Start
1
Init
2
Rekonstruktion
3
Aufruf
4
Ende

Fig. 1a

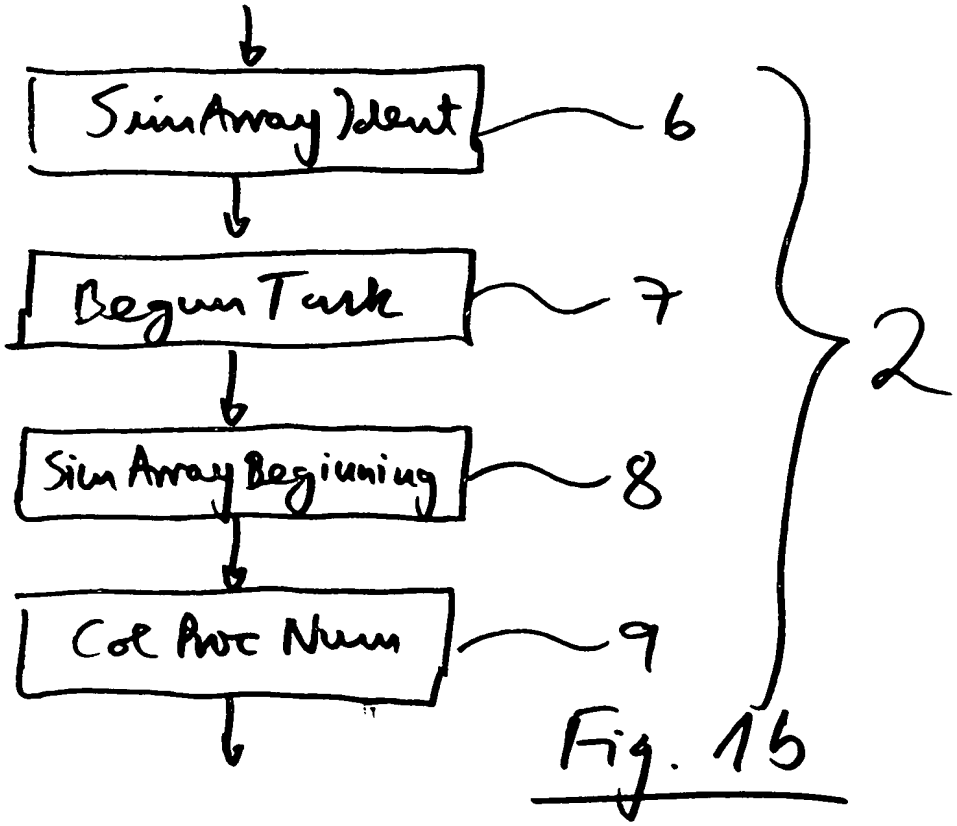

SimArrayIdent
6
BegunTask
7
SimArrayBeginning
8
ColRowNum
9
2

Fig. 1b

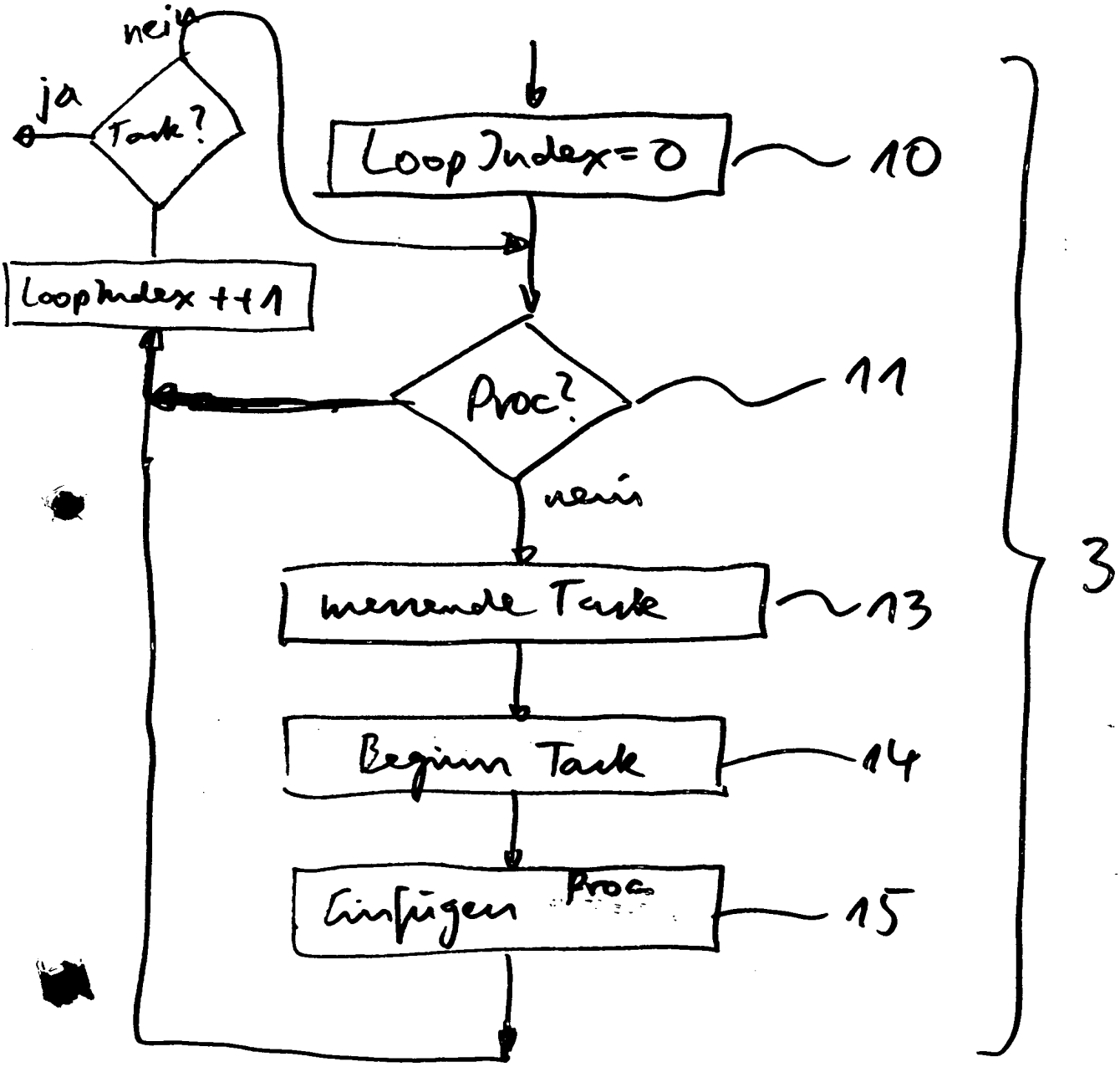

nein
ja
Task?
Loop Index = 0
10
Loop Index ++1
Proc?
11
nein
Merke Task
13
Begin Task
14
Einfügen Proc
15
3

Fig. 1c

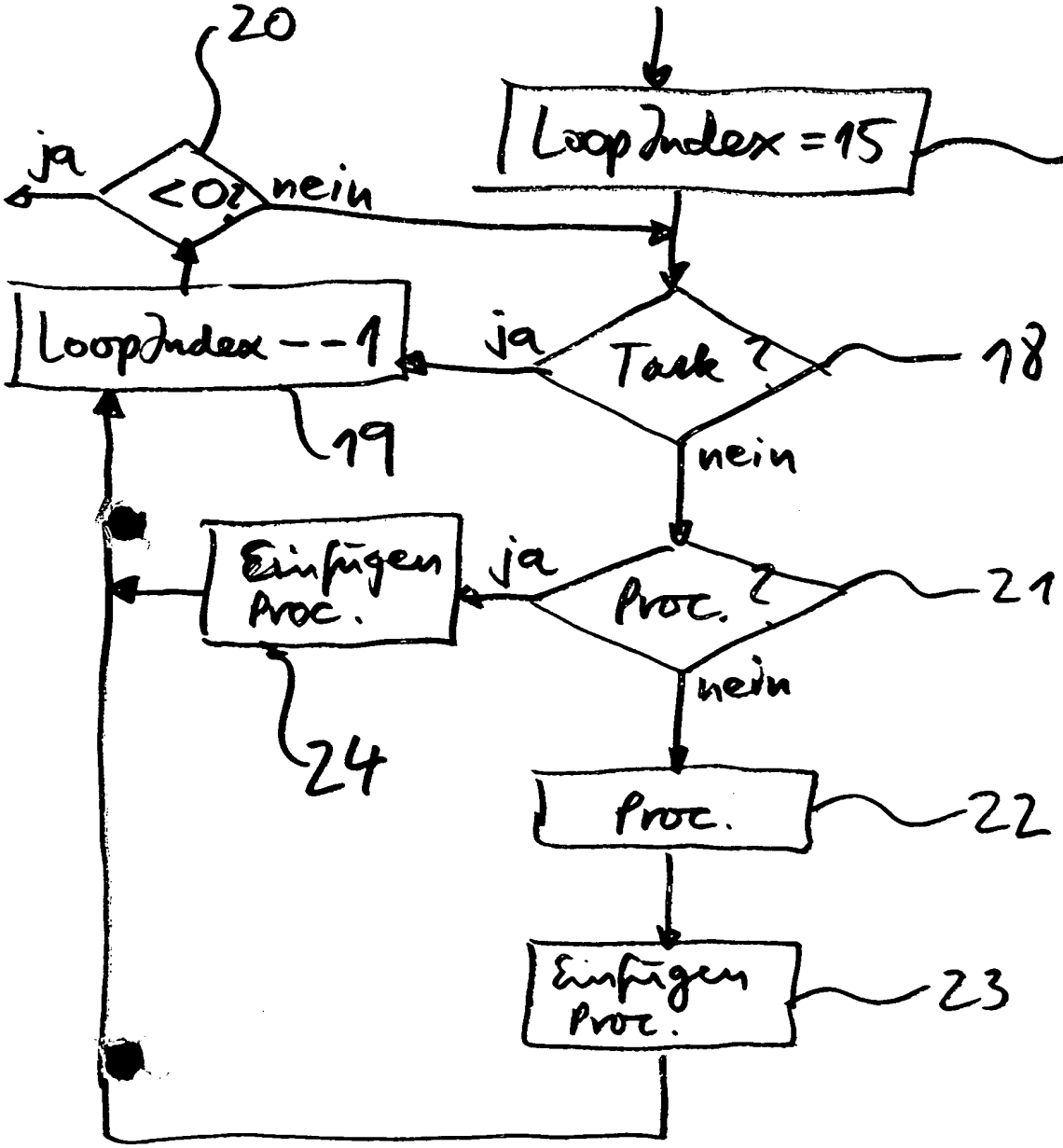

20
ja
<0?
nein
LoopIndex = 15
17
LoopIndex --1
ja
Task ?
18
19
nein
Einfügen Proc.
ja
Proc. ?
21
24
nein
Proc.
22
Einfügen Proc.
23

Fig. 1d

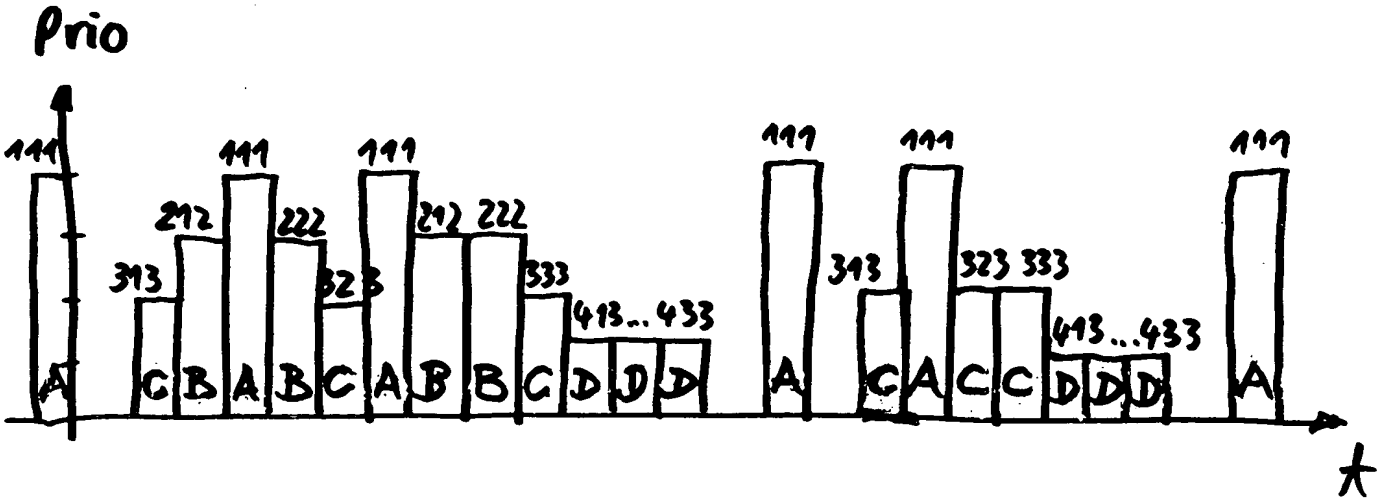

Prio
111
313
212
111
222
323
111
212
222
333
413...433
111
313
111
323
333
413...433
111
A
C
B
A
B
C
A
B
B
C
D
D
D
A
C
A
C
C
D
D
D
A
t

Fig. 2

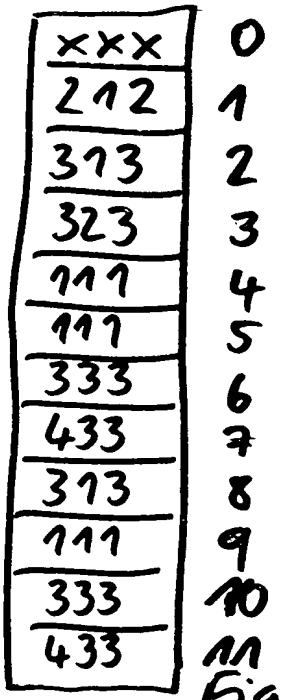

xxx
212
313
323
111
111
333
433
313
111
333
433
0
1
2
3
4
5
6
7
8
9
10
11

Fig. 3

A	0
A	1
B	2
A	3
B	4
C	5
D	6
A	7
A	8
C	9
D	10
A	11

Fig. 4

A	B	C	D
0	0	0	0

Fig. 7

0	1	2	3	4	5	6	7	8	9	10	11
111	111	222	111	222	333	433	111	111	333	433	111

Fig. 5

111	111	222	111	222	333	433	111	111	333	433	111
	212		323					313			
	313										

Fig. 6

R.39072

0	1	2	3	4	5	6	7	8	9	10	11
1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0

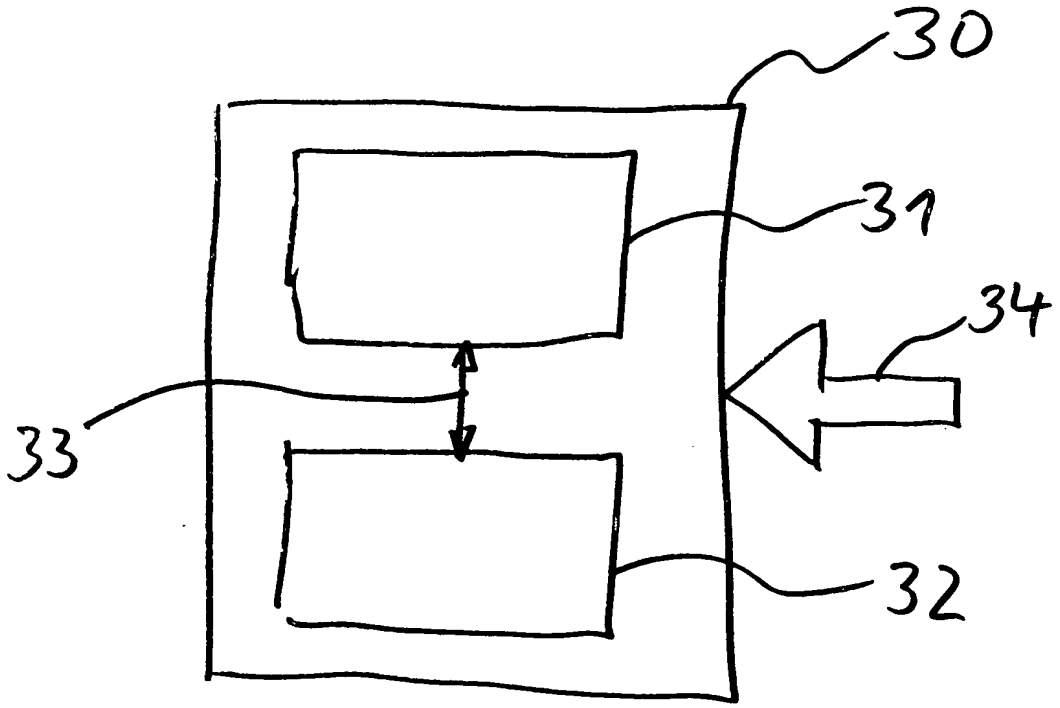
Fig. 8

111	0
313	1
212	2
111	3
222	4
323	5
111	6
222	7
333	8
433	9
111	10
313	11
111	12
333	13
433	14
111	15

Fig. 9

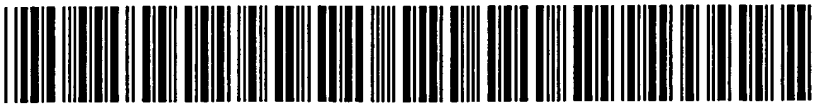
0	111	323	16
1	313	333	17
2	212	413	18
3	111	423	19
4	222	433	20
5	323	111	21
6	111		
7	212		
8	222		
9	333		
10	413		
11	423		
12	433		
13	111		
14	313		
15	111		

Fig. 10



Figur 11

US 1003454608P1



Creation date: 11-19-2004
Indexing Officer: BOO - BO OO
Team: OIPEBackFileIndexing
Dossier: 10034546

Legal Date: 08-26-2004

No.	Doccode	Number of pages
1	IDS	3

Total number of pages: 3

Remarks:

Order of re-scan issued on .....